INFORMATION TECHNOLOGY

Class – XII, Subject Code – 802

(2025-2026)

Rajbir Sahoo

First Release : 16th September 2025.

First Revised : 26th November, 2025.

Second Revised : 3rd December, 2025.

Newest Revised : 18th January, 2026.

Predictive Analytical Report on Information Technology (Subject Code 326/802) Examination Structure (Based of Algorithm on papers from 2025,2024,2023,2022)

I. Algorithmic Assessment of Examination Weightage

The maximum theoretical knowledge weightage in the Subject Specific Skills (Part B) is unequivocally assigned to **Database Concepts**.

Unit/Section	Maximum Marks (Part B: Subject Specific Skills)	Relative Weight (Approx.)
Database Concepts – RDBMS Tool	25 Marks	50%
Operating Web Based Applications	15 Marks	30%
JAVA - Fundamentals of Java Programming	10 Marks	20%
Employability Skills (Part A)	10 Marks	N/A (Mandatory Section)

The prediction vector confirms that **mastery of SQL/RDBMS syntax and theory is paramount** for securing the largest segment of marks.

II. Frequency Assessment and Repeated Topics

The analysis identifies critical, high-frequency topics appearing consistently across both objective (1-mark) and short/long subjective (2, 3, 4-mark) question formats.

A. Database Management System (RDBMS/SQL)

This module demonstrates the highest density of essential terminology and practical application questions.

High-Frequency Topic	Observed Assessment Mode	Key Concepts/Commands Assessed
Relational Model Terminology	Objective (Direct recall)	Defining **Tuple** (row), **Relation** (table), **Cardinality** (number of rows), and the purpose of the **NULL** value.
SQL Data Constraints	Objective & Short Answer	Usage and definition of `NOT NULL`, `DEFAULT` (e.g., setting a default age of 0), and `CHECK` (restricting values within a range).
SQL Data Types	Objective (Validation)	Differentiation between **`CHAR(n)` and `VARCHAR(n)`**. Identifying invalid data types (e.g., `STR(n)`).
Basic SQL Retrieval (`SELECT`)	Objective (Syntax recall) & Subjective (Query writing)	Correct syntax for retrieving all data (`SELECT * FROM...`). Use of **`DISTINCT`** keyword to suppress repetition. Filtering logic using `WHERE` and Boolean operators (`AND`, `OR`).
Advanced SQL Queries	Subjective (4 Marks - Practical application)	Application of the **`LIKE`** operator with wildcards (% for zero or more characters, _ for a single character). Use of **`ORDER BY`** clause.
DML/DDL Command Classification	Objective & Short Answer	Distinguishing non-DML commands (`CREATE`) from DML commands (`INSERT`, `UPDATE`, `DELETE`). Syntax for `INSERT` and structure modification (`ALTER TABLE... ADD/DROP`).
Aggregate Functions	Subjective (3 Marks - Explanations)	Definition and use of functions like `MAX()`, `MIN()`, `AVG()`. The distinction and purpose of the **`HAVING` clause** versus the `WHERE` clause.

B. Java Programming Fundamentals

The Java segment heavily emphasizes practical coding syntax, execution flow control, and core library classes.

High-Frequency Topic	Observed Assessment Mode	Key Concepts/Commands Assessed
Control Flow Structures	Objective & Subjective (Code prediction/writing)	Understanding and differentiating `while` vs. `do-while` (exit-controlled loop property). Use of the **`switch`** statement for multi-value matching, including conversion from `if-else` blocks.
String Manipulation	Objective (Method recall) & Subjective (Code writing/output prediction)	Knowledge of methods: **`length()`**, `toUpperCase()`/`toLowerCase()`, `indexOf()`, and `contains()`. Concatenation using the `+` operator.
Arrays and Libraries	Objective (Method recall/syntax) & Subjective (Definition)	Defining an **Array**. Correct syntax for array declaration and initialization. Use of `Arrays` class methods like `sort()` and **`binarySearch()`**.
OOP Core Concepts	Objective (Terminology) & Subjective (4 Marks - Explanation)	Defining **Object** and **Method**. Explaining **Access Modifiers** (especially `private`).
Exception Handling & Threads	Short Answer (Definition/purpose)	Purpose of `try` and **`catch`** blocks. Listing methods for **creating Threads**.

C. Employability Skills and Spreadsheets (ICT)

In Employability Skills, focus is divided between **Attitude/Motivation** and core **Spreadsheet** terminology.

High-Frequency Topic	Observed Assessment Mode	Key Concepts/Commands Assessed
Entrepreneurial Attitudes	Objective & Short Answer	Definition of entrepreneur/entrepreneurship. Core competencies like **Perseverance**, Initiative, and Decisiveness. Benefits of goal setting.
Self-Management/	Objective & Short Answer	Importance and ways to maintain **Positive Attitude**. Defining **Stress**.

Communication		Stages/factors of **Active Listening**.
Spreadsheet Concepts (ICT)	Objective (Terminology) & Short Answer (Steps)	Differentiating **Workbook** vs **Worksheet**. Identifying the **Cell**. Steps to apply **Filter** or **Save** a presentation.
Green Skills/Jobs	Short Answer (Definition/Examples)	Definition of a **Green Collar Worker**. Ways to reduce greenhouse gas emissions.

D. Operating Web Applications

High-Frequency Topic	Observed Assessment Mode	Key Concepts/Commands Assessed
E-Governance	Objective (Portal identification/definition)	Identifying government portals (e.g., `www.india.gov.in`). Knowledge of ICT-enabled services offered by the government (e.g., passport application, land records, voter ID). **NICNET** importance.
Web Application Development Lifecycle	Objective (Phase identification)	Understanding which tasks (e.g., fixing bugs/validation, establishing backend/frontend connectivity, feasibility checks) occur in the **Testing** or **Implementation** phase.
Online Learning/Webinar	Short Answer (Benefits/Difference)	Defining **Webinar**. Listing benefits of **online learning/tutorials**.

III. Algorithmic Network Prediction: Most Likely Asked Topics

The predictive model prioritizes topics that are **complex (require synthesis)**, **recursive (repeated across papers)**, and involve **applied programming/query tasks**.

1. Applied SQL Query Writing (High-Yield 4-Mark Questions)

The exam pattern consistently features large, multi-part subjective questions based on a given data table (e.g., `ACTIVITY`, `GARMENT`, or `TravelAgency`).

Predicted High-Yield Queries:

- **Data Aggregation and Grouping:** Writing a query using `GROUP BY` and the constraint `HAVING` (e.g., finding the sum of prize money per stadium, or counting groups exceeding a certain criteria).
- **Schema Manipulation:** Commands to **create a complex table** with multiple constraints (`PRIMARY KEY`, `NOT NULL`, Foreign Keys, `DEFAULT`), and commands to alter the table structure (`ALTER TABLE... DROP COLUMN/ADD FIELD`).
- **Search and Pattern Matching:** Complex `SELECT` queries utilizing `LIKE` with both wildcards (% and _) combined with `WHERE` and `ORDER BY`.

2. Java Control Flow and Methods (High-Yield 4-Mark Questions)

Practical Java code questions are a staple of the subjective section, often worth 4 marks.

Predicted High-Yield Tasks:

- **Loop Differentiation and Implementation:** Writing small Java programs that explicitly require the use of specific loops (`for` for factorial, or predicting output for `do-while` loops). The theoretical **differentiation between Entry and Exit controlled loops** is a high-probability short-answer question.
- **String Class Application:** Writing multiple lines of Java code to perform several string operations on a single variable (e.g., convert case, find index, check content using `contains()`, replace characters).
- **Conceptual Definition Synthesis:** Explaining why Java is considered an **Object-Oriented Programming (OOP) language**, using examples of data members and member methods.

3. High-Value Definitions and Constraints (Short Answer)

These conceptual questions recur often and provide easy marks if definitions are precise.

- **DBA Role and Responsibility:** Definition and a key task of the Database Administrator (DBA).
- **Web Application Development Phases:** Enumerating and explaining the key stages: Requirement, **Design**, **Implementation**, and Testing, and specifying the task performed in one phase (e.g., connectivity in Implementation).
- **Java Environment:** Definition and advantage of **JVM (Java Virtual Machine)**.

IV. Strategic Preparation Guide for the Concerned Student

To achieve maximal scores (crack the paper), the student must adopt a strategy focusing on high-accuracy technical execution and robust conceptual recall.

A. RDBMS/SQL Mastery (The 50% Foundation)

1. **Mandatory Syntax Recall:** Memorize the precise syntax for DDL and DML commands (CREATE TABLE, ALTER TABLE, DROP TABLE, INSERT INTO). A single syntax error can nullify marks in practical questions.
2. **Constraint Application:** Understand not just what constraints are, but *how* they are implemented in the `CREATE TABLE` command (e.g., setting `PRIMARY KEY`, `NOT NULL`, and `DEFAULT` values like 40000 for Salary, or 0 for Age).
3. **The "Big Three" Query Tools:** Dedicate concentrated practice time to mastering the intricate combination of `WHERE`, `LIKE` (especially the `%` and _ wildcards), and the ordering via `ORDER BY` (ASC/DESC).
4. ***Scientific Insight: The `HAVING` vs. `WHERE` Distinction:*** The `WHERE` clause filters rows *before* grouping, while the `HAVING` clause filters results *after* grouping by the aggregate function. This conceptual difference is a high-yield subjective explanation.

B. Java Programming Precision

1. **Code Output Analysis (The Prediction Matrix):** Practice tracing the output of code snippets involving arithmetic operators (e.g., modulus `%` and combined assignments `+=`, `-=`) and control flow statements (`while`, `do-while`, `switch`). Pay close attention to the use of `break` within `switch` statements, as its absence leads to fall-through errors.
2. **Method Recall:** Be fluent in identifying and writing the invocation for String and Array manipulation methods (`length()`, `toUpperCase()`, `binarySearch()`).
3. **Foundation Definitions:** Ensure crystal-clear definitions for foundational terms like **Variable**, **Array**, and the distinction between the assignment operator (`=`) and the comparison operator (`==`).

C. Conceptual and ICT Topics

1. **E-Governance Recall:** Memorize the national portal (`india.gov.in`), the specific purpose of **IRCTC** (train ticket booking portal.....**PYQ**), and one or two core functions of e-Governance (e.g., RTI application, registration of certificates).
2. **Employability Core:** Focus definitions on the entrepreneurial ecosystem (e.g., Perseverance, Green Collar Worker) and the distinction between spreadsheet elements (Workbook vs Worksheet, Cell vs Column).
3. **Presentation/Spreadsheet Steps:** Be ready to list the sequential steps for basic ICT tasks, such as applying a **filter** in a spreadsheet or the stages for saving a presentation in LibreOffice Impress.

	UNITS	NO. OF HOURS for Theory and Practical	MAX. MARKS for Theory and Practical
Part A	**Employability Skills**		
	Unit 1 : Commmunication Skills- IV	10	2
	Unit 2 : Self- Management Skills - IV	10	3
	Unit 3 : ICT Skills - IV	10	1
	Unit 4 : Entrepreneural Skills - IV	15	3
	Unit 5 : Green Skills - IV	05	1
	Total	**50**	**10**

	UNITS	NO. OF HOURS for Theory and Practical		MAX. MARKS for Theory and Practical
Part B	**Subject Specific Skills**	**Theory**	**Practical**	**Marks**
	Unit 1 : Database Concepts – RDBMS Tool	30	45	15
	Unit 2 : Operating Web Based Applications	15	20	10
	Unit 3 : JAVA	30	50	20
	Unit 4 : Work Integrated Learning IT - DMA	10	10	5
	Total	**85**	**125**	**50**

Part C	**Practical Work**		
	Java Program		10
	SQL Queries (Table Creation + 5 Queries)		10
	Practical File		10
	Viva		5
	Total		**35**

Part D	**Project Work** (Any Application made using Java NETBEANS IDE)		5
	Total		**5**
	GRAND TOTAL	**260**	**100**

SYLLABUS

I. Database Management System (DBMS) Concepts

Data is a collection of **raw facts** (numbers, text, symbols), **“Information”** is produced by processing data. **(PYQ)**

A **Database** is a collection of related data that has been recorded, organized, and made available for searching. A database is a representation of some aspect of the real world (miniworld), designed, built, and populated with data for a specific purpose.

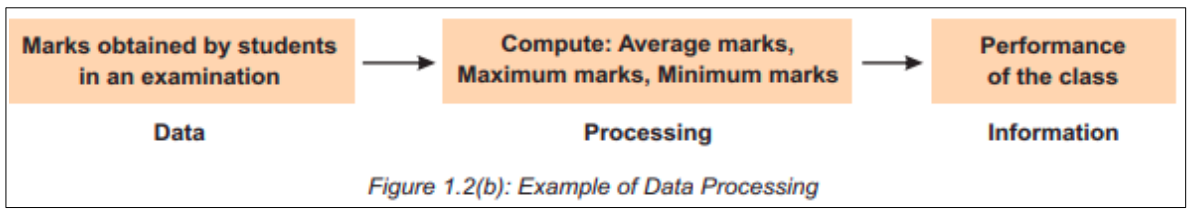

Figure 1.2(b): Example of Data Processing

A **Database Management System (DBMS)** is a collection of programs enabling users to create, maintain, and use a database. It facilitates the creation of a single repository of data that is defined once and accessed by different users according to their requirements. Examples of DBMS include **MySQL, Oracle, DB2, IMS, and IDS**.

Redundancy, **Inconsistency** and **Lack of data integration** are major problems in traditional file processing.

Database Operations The various operations performed on a database are:

1. **Defining the Database:** Specifying the data type and constraints on the data to be stored.
2. **Populating the Database:** Storing the data on a storage medium controlled by the DBMS.
3. **Manipulating the Database:** Modifying the database, retrieving data (querying), and generating reports.
4. **Sharing the Database:** Allowing multiple users simultaneous access.
5. **Protecting the Database:** Enabling protection from software/hardware failures and unauthorized access.
6. **Maintaining the Database:** Adapting to changing requirements easily.

   Mnemonic: “**DPS M**ein **P**re**M**”

Characteristics of DBMS The main characteristics of a DBMS are:.....**(PYQ)**

- **Self-describing Nature:** A DBMS stores not only the database but also the description of the data, known as **meta-data**, in a database catalogue or data dictionary. It contains structure of data and the constraints on the data.
- **Insulation Between Programs and Data:** Changes in data structure are recorded in the catalogue, meaning programs accessing the data do not need modification. This property is called **Program-Data Independence**.
- **Sharing of Data:** A multiuser environment allows simultaneous access, requiring the DBMS to include **concurrency control software** to prevent inconsistency problems.

- Mnemonic for Characteristics of DBMS: **SIS** (**S**-Self describing nature, **I**- Insulation b/w program and data, **S**- Sharing of Data)

Advantages of using a DBMS Approach (**PYQ**, As “Mention any two major significances of Database Management System.” 2 marks, **Sample paper** 3 marks 3 points) The database approach overcomes problems found in traditional file processing (like data redundancy, data inconsistency, and lack of data integration). Advantages include:

- **Reduction in Redundancy:** Data is concise as it is stored in a central repository, reducing storage cost.
- **Improved Consistency:** Inconsistency is reduced because there is a single copy of data for all users.
- **Improved Availability:** Information is easily shared among various users.
- **Improved Security:** Access can be restricted using passwords and access rights, typically controlled by the DBA.
- **User Friendly:** Easy access, modification, and deletion of data reduces dependency on computer specialists.

  Mnemonic: **US CAR** (**U**- User Friendly, **S**- Security, **C**- Consistency , **A**- Availability, **R**- Redundancy Reduce)

Limitations of using a DBMS Approach The two main disadvantages are:

1. **High Cost:** Implementation is costly and time-consuming, involving analysis, design, programming, and training.
2. **Security and Recovery Overheads:** Unauthorized access poses a threat, and data must be regularly backed up to prevent loss.

Types of Users of DBMS

1. **End Users:** Use the database for querying, modifying, and generating reports, without concern for its design or working.
2. **Database Administrator (DBA):** Administers the database and DBMS, responsible for authorization, monitoring use, technical support, and acquiring resources.....(**PYQ(2)**, Role of DBA)
3. **Application Programmers:** Write application programs, often using high-level languages and SQL.....(**PYQ,** _____ is a language that is used to manage data stored in an RDBMS., ans -> SQL), to interact with the database.
4. **System Analyst:** Determines end-user requirements and develops specifications, playing a major role in database design and assessing technical, economic, and feasibility aspects. **(PYQ)**

II. Relational Database Model

The **Relational Database** model, developed by **E.F. Codd** in **1970**, organizes data as a collection of relations, where each relation corresponds to a table of values. A **Relational DBMS (RDBMS)** is a DBMS based on this model, such as **Oracle**, **MySQL**, and **IBM DB2**.

Relational Terminology

- **Relation:** A table.....**(PYQ)**
- **Tuple:** A row in a table. **(PYQ)**
- **Attribute:** A column in a table.

- **Domain:** The data type of values in each column, representing the set of all possible atomic (indivisible) values that attribute may contain.
- **Degree:** The number of attributes (columns) in a relation.....**(PYQ)**
- **Cardinality:** The number of tuples (rows) in a relation.....**(PYQ)**
- **Relation Schema R:** Denoted by R (A_1, A_2, A_3, ... A_n), where R is the relation name and A_1, A_2, ..., A_n is the list of attributes.
- **Relation State:** The set of tuples in the relation at a specific point in time.

If you say row as **Tuple** →then column must be referred as **Attribute**

If you say row as **Record** → then column must be referred as **Field**

If you say row as **Row** → then column must be referred as **Column**

Characteristics of Relations

1. **Ordering of tuples** is not important.
2. **Ordering of attributes** is also unimportant.
3. **NO** two tuples should be identical **(PYQ, T or F)** (values in at least one column must be different).
4. Values in each tuple are **atomic** (indivisible).
5. A special value called **NULL** is used if an attribute value is unknown, not applicable, or unavailable. **(PYQ)**

IMPORTANT PYQ

Q) Jatin has given the following query for the table Travel to display the

names of customers whose age is not specified, but he is unable to execute it successfully. Write the correct query after rectifying the error.

```
Select Name, From Travel where Age = 'NULL';
```

Answer:

```
SELECT Name
FROM Travel
WHERE Age IS NULL;
```

III. Relational Model Constraints

Constraints are restrictions on the values stored in a database. There are several types of constraints in the Relational model:

1. **Domain Constraint:** Specifies that the value of every attribute in each tuple must belong to that attribute's defined domain.....(**PYQ**) (e.g., Employee_ID must be a 4-digit number).
2. **Key Constraint:** Involves identifying unique identifiers:
   - **Superkey:** A set of one or more attributes where no two tuples in a relation state have the same combination of values. The combination of all attributes is always a superkey.
   - **Key:** The **minimal superkey**, meaning removing any attribute (column) causes it to not being a superkey.
   - **Candidate Key:** A key as described above.
   - **Primary Key:** One candidate key designated to identify tuples in a relation. It is preferably the one with the least number of attributes, and is usually underlined in the schema.
   - **Foreign Key:** Foreign Key in relation R1 is a set of attributes that refers to the primary key in another relation R2 (the referenced relation). The value of the foreign key must either occur as a primary key value in some tuple of R2 or be NULL.
3. **Null Value Constraint:** Requires certain attributes to be constrained as **NOT NULL** (while input writing, eg First_Name VARCHAR(20) NOT NULL....), meaning they cannot have null values.
4. **Entity Integrity Constraint:** Specifies that the **primary key of a relation cannot have a null value**. This is because allowing NULL values for a primary key would violate the requirement that primary keys contain no duplicates.

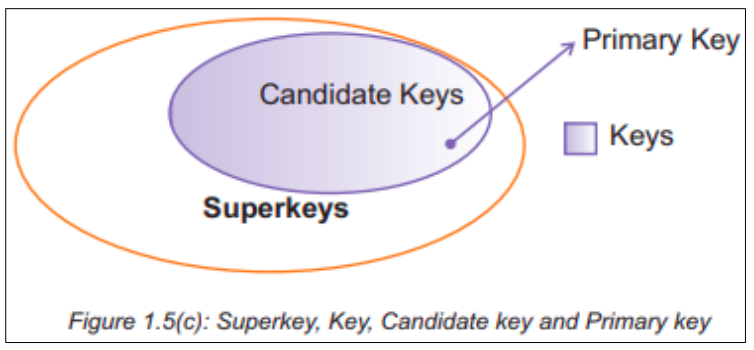

Figure 1.5(c): Superkey, Key, Candidate key and Primary key

5. **Referential Integrity Constraint:** This constraint links two relations using a **Foreign Key**. The main purpose of this constraint is to check that data entered in one relation is consistent with the data entered in another relation. (**PYQ**, Define Referencial Integrity)

Referential integrity can enforce actions when a change occurs in the referenced table (R2), using **ON DELETE** and **ON UPDATE** clauses with actions such as:

- **SET NULL:** Sets the corresponding foreign key attribute in the referencing table (R1) to NULL.
- **CASCADE:** Deletes or updates corresponding tuples in the referencing table (R1).
- **RESTRICT:** Rejects the delete or update operation on primary key table if related foreign key values exist in the referencing table (R1).

Eg:-

```
•  FOREIGN KEY (Dept_No) REFERENCES Department (Dept_ID) ON DELETE SET NULL ON UPDATE SET NULL
•  FOREIGN KEY (Dept_No) REFERENCES Department (Dept_ID) ON DELETE CASCADE ON UPDATE CASCADE
```

Database Constraints:

1. NOT NULL

- Used when an attribute **cannot be left empty**.
- Example: Teacher's *First_Name* cannot be NULL.
- Table example:

```
CREATE TABLE TEACHER(
  Teacher_ID INTEGER,
  First_Name VARCHAR(20) NOT NULL,
  Last_Name VARCHAR(20),
  Gender CHAR(1),
  Salary DECIMAL(10,2),
  Date_of_Birth DATE,
  Dept_No INTEGER
);
```

2. DEFAULT.....(PYQ)

When user does **not enter a value**, DB uses the default.

- Example: Default Salary = **40000**.
- Table example:

```
CREATE TABLE TEACHER(
  Teacher_ID INTEGER,
  First_Name VARCHAR(20) NOT NULL,
  Last_Name VARCHAR(20),
  Gender CHAR(1),
  Salary DECIMAL(10,2) DEFAULT 40000,
  Date_of_Birth DATE,
  Dept_No INTEGER
);
```

3. CHECK.....(PYQ)

- Restricts attribute values to a **range/condition**.
- Example: Dept_No must be ≤ **110**.
- Table example:

```
CREATE TABLE TEACHER(
  Teacher_ID INTEGER,
  First_Name VARCHAR(20) NOT NULL,
  Last_Name VARCHAR(20),
  Gender CHAR(1),
  Salary DECIMAL(10,2) DEFAULT 40000,
  Date_of_Birth DATE,
  Dept_No INTEGER CHECK (Dept_No <= 110)
);
```

MySQL does **NOT** support CHECK constraint, although it will not give any error if you have a check constraint in your table.

Extra Info: Only MySQL versions less than 8.0.16 dont support CHECK constraint.

CHECK can also compare two attributes of same table. Example :-

```
CREATE TABLE Student (
  Marks_Obtained INT,
  Max_Marks INT,
  CHECK (Marks_Obtained <= Max_Marks)
);
```

4. KEY CONSTRAINTS (PRIMARY KEY)

a) Single-Attribute Primary Key

- Example: Teacher_ID is primary key.

```
CREATE TABLE TEACHER(
  Teacher_ID INTEGER PRIMARY KEY,
  First_Name VARCHAR(20) NOT NULL,
  Last_Name VARCHAR(20),
  Gender CHAR(1),
  Salary DECIMAL(10,2) DEFAULT 40000,
  Date_of_Birth DATE,
  Dept_No INTEGER
);
```

Notes:

- Primary key automatically means **NOT NULL** is present.
- DESC shows **PRI** under Key column

b) Composite Primary Key

- When primary key has **more than one attribute**.
- Example: Primary Key = (Teacher_ID, Date_of_Birth)

```
CREATE TABLE TEACHER(
```

```
  Teacher_ID INTEGER,
  First_Name VARCHAR(20) NOT NULL,
  Last_Name VARCHAR(20),
  Gender CHAR(1),
  Salary DECIMAL(10,2) DEFAULT 40000,
  Date_of_Birth DATE,
  Dept_No INTEGER,
  PRIMARY KEY (Teacher_ID, Date_of_Birth)
);
```

5. REFERENTIAL INTEGRITY (FOREIGN KEY)

Concept

- FOREIGN KEY in one table refers to PRIMARY KEY in another.
- Example setup:
    - Department(Dept_ID, Dept_Name)
    - Teacher(… , Dept_No)

Dept_No is foreign key referencing Dept_ID

- First create Department table:

```
CREATE TABLE Department(
  Dept_ID INTEGER PRIMARY KEY,
  Dept_Name VARCHAR(20) NOT NULL
);
```

Then Teacher table:

```
CREATE TABLE Teacher(
  Teacher_ID INTEGER PRIMARY KEY,
  First_Name VARCHAR(20) NOT NULL,
  Last_Name VARCHAR(20),
  Gender CHAR(1),
  Salary DECIMAL(10,2) DEFAULT 40000,
  Date_of_Birth DATE,
  Dept_No INTEGER,
  FOREIGN KEY (Dept_No) REFERENCES Department(Dept_ID)
);
```

Alternatively,

```
CREATE TABLE Teacher(
Teacher_ID INTEGER PRIMARY KEY,
First_Name VARCHAR(20) NOT NULL,
Last_Name VARCHAR(20),
Gender CHAR(1),
Salary DECIMAL(10,2) DEFAULT 40000,
Date_of_Birth DATE,
Dept_No INTEGER REFERENCES Department(Dept_ID));
```

Extra notes

- In DESC output, foreign key column shows **MUL** → means values can repeat.
- Referenced table (The table whose primary key is being pointed to by a foreign key in another table) **must be created first**.
- Actions like ON DELETE / ON UPDATE can be specified.

IV. Structured Query Language (SQL)

SQL is the language used to manage data stored in an RDBMS. It is composed of:

- **Data Definition Language (DDL):** Used to define the structure and constraints of data. E.g – CREATE.....(**PYQ(2),** Asked what category)
- CREATE, ALTER, DROP,RENAME,TRUNCATE,DESC (Mnemonic: Crazy Angry Dragons Ride Tricky Donkeys)
- **Data Manipulation Language (DML)(PYQ : Defn + examples):** Used to insert, modify, and delete data in a database. E.g - INSERT, UPDATE, DELETE, SELECT (Mnemonic: Some Ugly Dogs Itch)

SQL typically uses the terms **table, row, and column** for the relational model terms **relation, tuple, and attribute**, respectively.

Key SQL Commands (DDL)

Command	Purpose	Syntax/Usage Notes
CREATE TABLE	To create a new table/relation.	Specifies column name, data type, and optional constraints. Commonly used data types include `CHAR(n)`, `VARCHAR(n)`, `DATE`, `INTEGER`, and `DECIMAL(m, d)`.
CREATE DATABASE	To create a new database.	`CREATE DATABASE School;`.
USE (PYQ)	To select the database for subsequent operations.	`USE School;`.
SHOW DATABASES/TABL	To list existing databases or tables in the current	`SHOW DATABASES;` or `SHOW TABLES;`.

ES	database.	
DROP DATABASE/TABLE	To delete a database or table.....**(PYQ)**	Can include `CASCADE` or `RESTRICT` options (though not supported in MySQL 5.6.20 for tables).
DESC (or DESCRIBE)	Displays the structure and description of a table. **(PYQ)**	`DESC Teacher;`.
ALTER TABLE	Used to modify the base table definition.	Can add a column (`ADD Age INTEGER`), drop a column (`DROP Dept_No CASCADE`), or alter a column (`ALTER Salary SET DEFAULT 30000`). It is also used to drop or add constraints.

Key SQL Commands (DML)(PYQ)

Command	Purpose	Usage Notes/Clauses
INSERT	Used to insert a tuple (row) into a table.	Values must be specified in the correct order, unless attributes are explicitly listed. If a value is not specified, its `DEFAULT` value is used, or `NULL` otherwise.
UPDATE	Used to update attribute values in one or more tuples.	Requires a `SET` clause and typically a `WHERE` clause to specify which rows to modify. Algebraic expressions can be used in the `SET` clause.
DELETE (PREBOARDS)	Used to delete one or more tuples from a table.	The `WHERE` clause specifies which tuples to delete; omitting it deletes *all* tuples.
SELECT	Used to retrieve information from a database.	Basic structure: `SELECT <attribute list> FROM <table list> WHERE <condition>`.

CHEAT

```
DROP TABLE Teacher RESTRICT;
```
What this means ↑ : "Only delete this table **if nothing else is using it**"

```
DROP TABLE Teacher CASCADE;
```
What this means ↑ : "Also remove **anything else that depends on it**"

In MySQL server 5.6.20, you can simply type Drop Table followed by the table name to delete a table from the database. However, it will **NOT** allow you to drop a table if the table to be deleted is being referenced in some other table.

ALTER TABLE Uses:

(a) Adding a Column

To add a new column to an existing table.

```
ALTER TABLE Teacher ADD Age INTEGER;
```

(b) Dropping a column

A column can be dropped using this command but one **MUST** specify the options (RESTRICT or CASCADE) for the drop behavior

```
ALTER TABLE Teacher DROP Dept_No CASCADE;
```

NOTE: RESTRICT and CASCADE options are not supported in MySQL server 5.6.20.

(c) Altering a Column

A column definition can also be altered. For example – dropping the default value or defining a new default value.

```
ALTER TABLE Teacher ALTER Salary DROP DEFAULT;
ALTER TABLE Teacher ALTER Salary SET DEFAULT 30000;
```

(d) Dropping Keys

```
ALTER TABLE Teacher DROP FOREIGN KEY TEACHER_FK;
```

CASCADE or RESTRICT option can be specified

Primary key can be dropped by using the command:

```
ALTER TABLE Teacher DROP PRIMARY KEY TEACHER_PK;
```

However, primary key can**NOT** be removed if it is the **only** primary key of the table. (Will throw error)

(e) Adding a constraint

```
ALTER TABLE Teacher ADD CONSTRAINT TEACHER_FK FOREIGN KEY (Dept_No) REFERENCES
Department(Dept_ID) ON DELETE SET NULL ON UPDATE SET NULL;
```

IMPORTANT PYQ

QUESTION:

A query to modify the field of dept from "marketing" to "sales" would be?

WRONG

```
ALTER TABLE [table_name] MODIFY dept;
SET dept = "sales" WHERE dept = "marketing";
# (ALTER is a structure-change command, not for data)
```

CORRECT

```
UPDATE table_name
SET dept = 'sales'
WHERE dept = 'marketing';
# (UPDATE is used to change values)
```

- **WHERE:** Uses logical comparisons (`=, <, >, <=, >=, <>`) and Boolean operations (`AND`, `OR`) to filter rows.
- **Cross Join:** Occurs when joining multiple tables without a `WHERE` clause, resulting in the **Cartesian product** of the rows. **.....** (**PREBOARD** Question)

Example:

Table Student

Student_ID	Name
1	A
2	B

Table Subject

Subject_ID	Subject
101	Math
102	Physics

```
SELECT *
FROM Student, Subject;
```

Result

Student_ID	Name	Subject_ID	Subject
1	A	101	Math
1	A	102	Physics
2	B	101	Math
2	B	102	Physics

- **Join Condition:** Used in the `WHERE` clause to combine records from two tables based on column equality (e.g., `Dept_ID=Dept_No`).
- **Aliases (AS):** Used to rename columns or resolve ambiguity when table names or attribute names conflict.
- **DISTINCT:** Keyword used to list unique (non-repeated) values of an attribute in the result set. **(PYQ(2))**
- **LIKE:** Used for string pattern matching, employing `%` (zero or more random characters) (**PYQ** '%')and _ (a single character).
- **ORDER BY:** Sorts the query results, defaulting to ascending order (`ASC`) **(PYQ)** but allowing descending order (`DESC`).
- **IS NULL / IS NOT NULL:** Used to test for unknown, unavailable, or non-applicable values in a column.
- **Nested Queries (Subqueries):** A query within the `WHERE` clause of another query, often using the `IN` keyword.

```
SELECT Tname
FROM Teacher
WHERE Deptno IN (4, 7);
```

Aggregate Functions Mathematical functions applied to groups of values.....(**PYQ**, Fill in the blanck.....The functions that are used to apply certain mathematical functions on a group of values in a database are called _____.) (**PYQ** : Explain the use of any three aggregate functions in SQL.)

- **COUNT:** Counts the number of tuples or rows.
- **SUM:** Finds the total sum of values for a numeric attribute.
- **MAX / MIN:** Finds the maximum or minimum value for a numeric attribute.
- **AVG:** Finds the average value for a numeric attribute. **(PYQ)**
- **GROUP BY:** Groups the result set based on a specified attribute. (**PYQ:** Explain the GROUP BY clause in SQL with an example.)
- **#IMP :- HAVING:** Applies a condition to the group defined by `GROUP BY` (unlike `WHERE`, which filters individual rows)..... (**PYQ**, What is the purpose of **HAVING** clause ? How is it different from **WHERE** clause in SQL ? Give example to support your answer. **4 MARKS**)

Example:-

```
-- Using WHERE (filtering rows)
SELECT * FROM STUDENT
WHERE MARKS > 80;

-- Using HAVING (filtering groups)
```

```
SELECT CLASS, AVG(MARKS)
FROM STUDENT
GROUP BY CLASS
HAVING AVG(MARKS) > 75;
```

V. Database Applications and Connectivity

Potential Work Areas for DBMS Applications Database management systems are essential across various domains, including:...(**PYQ** : Name any two domains where database applications may be used.)

The sources identify several key domains where Database Management Systems (DBMS) are essential for efficiency:

- **Education:** Managing student details, marks, results, faculty records, and school infrastructure like department and course offerings. (**PYQ(2),** Specifically)
- **Banking:** Storing personal customer data (address, age, PAN), account and loan information, and daily transaction histories. **(PYQ)**
- **Hospitals:** Maintaining patient health records, hospitalization dates, doctor/staff details, and inventory for medical equipment and medicines. **(PYQ)**
- **Government Sector:** Managing electoral rolls, various taxes (Income, Sales, House), criminal records, and national registrations like AADHAR or vehicle permits. **(PYQ)**
- **Companies & E-Commerce:** Tracking employee payroll, project progress, infrastructure investments, product inventories, and customer orders.
- **Airlines & Railways:** Storing flight/train schedules, fares, passenger capacity, and managing both online and offline reservations.
- **Telecommunications:** Managing communication networks, customer call records, billing, and subscription plans.
- **Hotels:** Tracking guest check-in/out times, room availability, staff details, and menu items. **(PYQ, 3 MARK!)**

3. Case Study: XYZ Shopping Website

This application serves as a practical example of a 24/7 online shopping portal.

- **Website Features:** Users can browse various categories (Books, Clothes, Electronics), add items to a **shopping cart**, and perform **comparative analysis** of products before purchasing.
- **Payment & Delivery:** The system supports multiple payment modes, including **Cash on Delivery (COD)**, credit/debit cards, and net banking, followed by delivery to the customer's specified address.

4. Database Entities and Schema (Detailed)

The application requires several interconnected tables, each defined by a **Primary Key (PK)** and often utilizing **Foreign Keys (FK)** for relationships:

- **CATEGORY:** Stores `Category_id` (PK), `Name`, and `Description`.
- **PRODUCT:** Stores `Product_id` (PK), `Name`, `Category_id` (FK referencing Category), `Price`, `Quantity`, `Discount`, `Brand`, `Color`, `Size`, `Seller_id` (FK), and `Description`.
- **CUSTOMER:** Maintains user profiles with `Customer_id` (PK), `Password`, `First_name`, `Last_name`, `Address`, `Email_id`, and `Contact_num`.
- **SELLER:** Records `Seller_id` (PK) and the seller's `Rating`.
- **WISH_LIST:** Uses a **composite Primary Key** (`Customer_id` and `Product_id`) to track items users want to buy later.
- **ORDER:** Tracks transactions via `Order_id` (PK), `Customer_id` (FK), `Product_id` (FK), `Shipment_id` (FK), `Order_date`, and `Current_status` (Pending/Delivered).
- **SHIPMENT:** Stores `Shipment_id` (PK), `Details` (location/dispatch name), and `Shipment_date`.
- **PAYMENT:** Records `Payment_id` (PK), `Order_id` (FK), `Payment_amount`, `Payment_date`, and `Mode` (Card, Net Banking, COD).

5. System Functionality and Logic

The system uses Java classes to manage these tables through specific routines:

- **Category/Product Management:** Classes like `Category` and `Product` use functions such as `insert`, `delete`, `modify`, and `display` to manage the inventory available on the site.
- **Seller Management:** Handles seller data and ratings; if a product is offered by multiple sellers, a composite key (`Product_id` and `Seller_id`) is used to manage the listings.
- **Customer Interaction:** Existing users log in through a specialized interface, while new users use a **Sign Up** interface that triggers the `insert_customer` function to save their data for future orders.
- **Order Completion Sequence:**
  1. Add items to the cart.
  2. Select the "Buy" option.
  3. Log in or provide shipping details.
  4. Choose a payment option and complete the transaction.
- **Order and Payment Processing:**
  1. The `Order` class uses `insert_order` for placement and `display_order` for tracking status via the `order_id`.
  2. For **COD payments**, the `Payment_date` attribute is only updated once the product is physically delivered.
- **Shipment:** The `Shipment` class uses `insert_shipment_details` to record the logistics of the order.

Database Connectivity using Java

To connect a MySQL database to a Java program, classes from the **java.sql package** are required. The general steps involve:

1. **Registering the MySQL Server** in the NetBeans IDE Services tab.
2. **Adding the MySQL Connector JAR** file to the project Libraries.
3. Invoking the `getConnection()` method of the **DriverManager** class using the database URL, username, and password to establish a connection (e.g., `jdbc:mysql://localhost:3306/bookstore`).
4. Creating a **Statement** object using the `createStatement()` method of the Connection object.
5. Executing an SQL query using the `executeQuery()` method of the Statement object, which returns a **ResultSet** object.
6. Iterating through the rows of data returned by the query using the `next()` and `getString()` methods of the ResultSet object.
7. Wrapping these operations in a `try-catch` block to handle `SQLException`.

Imp Solved Questions

Q1) To list names of teachers in alphabetical order.

```
A) SELECT F_N, LN FROM TEACHER ORDER BY F_N, LN ASC;
```

IMPORTANT: THIS will make the DBMS sort F_N alphabetically, the LN is mentioned so that F_N is associated with its concerned LN; if no LN were mentioned then FN only sort and LN not sort accordingly!

Q2) To retrieve names and department numbers of all the teachers ordered by DEPT nos and within each dept ordered by names of teachers in desc order.

```
A) SELECT F_N, L_N, DEPT_NO FROM TEACHER ORDER BY DEPT_NO ASC, F_N DESC, L_N DESC;
```

INPUT:

F_N	L_N	DEPT_NO
John	Smith	2
Adam	Khan	1
Zara	Ali	1
John	Brown	2

OUTPUT:

F_N	L_N	DEPT_NO
Zara	Ali	1
Adam	Khan	1
John	Smith	2
John	Brown	2

Q3) To find total salary of all the teachers.

```
A) SELECT SUM(SALARY) AS TOTAL_SALARY FROM TEACHER;
```

Q4) To find max and min salary

```
SELECT MAX(SALARY) AS MAX_SALARY, MIN(SALARY) AS MIN_SALARY FROM TEACHERS;
```

Q5) To count no off teachers having earnings more than 40k

```
A) SELECT COUNT(SALARY) FROM TEACHER WHERE SALARY>40K
```

Q6) To recieve total no of teachers in “CSE” DEPT.

```
A) SELECT COUNT(*) AS KMB FROM DEPT, TEACHER WHERE DEPT_NAME = “COMPUTER_SCIENCE” AND DEPT_NO = DEPT ID;
```

Q7) each student’s total marks (sum of all their entries in the marks table).

```
A) SELECT student_id, SUM(marks) AS totalmarks FROM marks GROUP BY student_id;
```

INPUT

student_id	marks
1	50
1	30
2	40

student_id	marks
2	60
3	90

OUTPUT

student_id	totalmarks
1	80
2	100
3	90

NOTE: “each Student”, “each Station”, “each Stadium” etc give you clues that you need to SELECT [Student_ID/Stadium/Station_ID].....GROUP BY [Student_ID/Stadium/Station_ID]

QIf a category has many products, how should they be displayed to customers? **(CBSE SQP 2025-2026)**

A) Products can be displayed based on:

1. Popularity: Show the most viewed or purchased products first. Database should store view count or purchase count.
2. Price: Show products from low to high or high to low. Database should store price of each product.
3. Ratings/Reviews: Show highest-rated products first. Database should store customer ratings or review scores.

[MARKING SCHEME]

MAHA IMP

Q) Write a query to find the no of ticket machines in each station

```
A) SELECT Station_ID, COUNT(Machine_ID) AS Machine_No FROM Machine GROUP BY Station_ID:
```

Remember:- GROUP BY name has to be equal to the SELECT name. COUNT();

IMPORTANT PYQ

Consider the following table **ACTIVITY** and answer the following : *1+1+1+1*

Acode	Activity Name	Stadium	Participants_ Number	Prize_ Money	Schedule_Date
101	Relay 100 x 4	Star Annex	16	10000	23-Jan-2005
102	High jump	Star Annex	10	12000	12-Dec-2003
103	Shot Put	Super Power	12	6000	14-Feb-2004
105	Long Jump	Star Annex	12	10000	01-Jan-2004
108	Discus Throw	Super Power	10	15000	19-March-2004

(a) Write an SQL query to display the names of all activities with their Acodes in descending order.

(b) Write an SQL query to display sum of Prize Money for the Activities played in each Stadium separately.

(c) Write an SQL query to increase the prize money by **1000** for Activity named **Shot Put**.

(d) Give output for the following query :

```
SELECT MIN (Prize_Money) FROM ACTIVITY ;
```

ANSWER

```
(a) SELECT Activity_Name, Acode FROM ACTIVITY ORDER BY Acode DESC;
(b) SELECT Stadium, SUM(Prize_Money) FROM ACTIVITY GROUP BY Stadium;
(c) UPDATE ACTIVITY SET Prize_Money = Prize_Money + 1000 WHERE Activity_Name = 'Shot Put';
```

(d)

```
+------------------+
| MIN(Prize_Money) |
+------------------+
|             6000 |
+------------------+
1 row in set (0.00 sec)
```

Unit II: Operating Web

I. Operating Web-based Applications

- A **web-based application** is an application that can be accessed via a web browser over a network connection.
- Web-based applications provide the convenience of **availability 24x7 across the world**.
- Users can access these facilities via the Internet from various locations, such as home, office, or while traveling.

- Examples of uses include reservation of tickets and bookings, e-governance, online shopping with online payment, and learning through online tutorials and tests.
- Web-based online applications can be broadly categorized into two types: **(DPS PYQ)**
    1. **Applications requiring financial online transactions:** Examples include online reservation systems, online shopping, and bill payments.
    2. **Applications providing information and interaction:** Examples include e-governance and online tutorials and tests, which involve filling forms, posting queries, viewing information, sending email, or chatting.

II. Online Reservation Systems

- An **online reservation system** allows the booking or cancellation of tickets through the use of the Internet.
- A user can retrieve information and perform transactions related to ticket reservation.
- **Examples** of online reservation systems include Airline, Railway, Bus, Movie ticket booking, Hotel booking, Car rental, and Tour and travel booking.
- These systems store information about services (e.g., flight timings, prices, provider, seat availability) and allow users to check information and book tickets.

A. Advantages for Consumers

1. **Convenient (24x7 Access):** Users can book tickets anytime, anywhere, requiring only a computer, Internet access, and a payment card (credit, debit, etc.).
2. **Saves Time and Energy:** Users avoid going physically to a booking office, saving time spent traveling, finding parking, or standing in queues.
3. **Easy Modifications:** Changes like cancellation or upgradation are easy.
4. **Price Comparison:** Users can check and compare prices across multiple service providers (e.g., comparing air ticket providers on websites like `makemytrip.com`) to get the best deal.
5. **Security:** Information remains secure if the user's password is secure, as most sites require user profiles with usernames and passwords.
6. **Confirmation of Reservation:** Booking confirmation is fast and secure, typically taking only a few seconds, and the confirmed ticket can be viewed on mobile devices or printed.
7. **Making Choices:** Users can select specific arrangements (e.g., seat selection, lower/upper berth in railways, type of food, booking beddings) and read reviews posted by others.
8. **Discount Schemes:** Users can avail special offers like season, advance booking, or frequent user discounts.
(Prerequisite: The user must have an understanding of using the computer for reservation).

C-S-E-P-D

C → Convenient (24×7 access)

- **S → Saves time & energy**
- **E → Easy modifications** (cancel / upgrade)
- **P → Price comparison**
- **D → Discount**

B. Advantages for Providers (PYQ)

- **Requires Less Staffing:** Reduced manpower is needed as tasks are computerized.
- **No Spatial Restrictions:** The organization's physical office location does not matter to the customer, allowing the company to operate from anywhere (e.g., basement, garage, small office) and reducing office bills (rent, electricity), though initial website setup costs are incurred.
- **Global Access:** Reservation is available to anyone with Internet access, regardless of their physical location, expanding the scope of access.

C. Precautions while Performing Online Transactions (PYQ, asked one)

- Internet transactions are susceptible to fraud.
- User identification relies on username and password combination for authentication.
- Simple passwords can be easily cracked, leading to unauthorized access.
- **Recommended Actions for Safer Passwords:**
    - Make passwords **complex** (mix case, use numbers and special characters) to decrease vulnerability to automated attacks.
    - Be cautious not to leave passwords exposed or share them.
    - Never use names of near and dear ones as passwords.

D. Using Online Reservation Systems (General Steps, PYQ)

1. **Open the online reservation website:** Use a web browser (e.g., Internet Explorer, Google Chrome.....**(PYQ(2))**).
2. **Access the site:**
    - Type the address directly in the address bar, or
    - Use a search engine (e.g., `google.com`) to search for the site and get the address.
3. **Browse, search, and use the website for reservation**.

E. Case Study: Booking Rail Ticket (IRCTC).....

1. **Preparation:** Need a web browser and Internet connection.
2. **Access:** Type `https://www.irctc.co.in/` in the address bar.
3. **Login/Signup:** An account is required for booking.
    - **Existing User:** Type username and password to login.
    - **New User:** Click `<Signup>`, fill the individual registration form (mandatory fields marked with *), and click `<Submit>`.
4. **Book a Ticket:**
    - Log in to the account.
    - In the "Plan My Travel" box, enter start station, end station, date of travel, ticket type (e-ticket), and quota.
    - Click `<Find Trains>`.
    - A list of available trains, including number, name, departure/arrival times, travel distance/time, and seat category availability (e.g., 1A, 2A, 3A, SL) is displayed. (A cross 'x' depicts non-availability of a category).
    - Select the desired seat category for a train.
    - A screen displays seat availability for the requested date and a few days ahead, along with the ticket fare.
    - Click `<book>`.
    - A form appears: Enter passenger details (name, age, sex, berth/seat preference, senior citizen status).
    - Press `<enter>` to view booking details.
5. **Payment:** Choose a payment option (credit card, debit card, etc.) and enter details.
6. **Confirmation:** An electronic copy of the paid ticket is displayed on screen and emailed to the registered address.

III. E-Governance

- **E-Governance (Electronic Governance):(PYQ, Fill in the blancks keyword: E-governance)** The use of **Information and Communication Technology (ICT) tools** (Internet, LAN, mobiles, etc.....**(PYQ)**) by the government for its citizens.
- **Purpose/Goal:** Provides access to quality information and better services, empowers citizens socially and economically, and ushers **transparency** by enabling citizens to access information directly, without middlemen.
- **Examples of Services:** Registration of birth/death certificates, booking railway tickets, RTI application submission.

A. Initiatives in India

- **NICNET (1987):** National satellite-based computer network, providing the main thrust for e-Governance. **(PYQ)**
- **DISNIC (District Information System of the National Informatics Centre):** Program to computerize all district offices, offering free hardware and software.
- NICNET expanded to all district headquarters via State capitals by 1990.
- **National e-Governance Plan (NeGP) (2006):** Formulation by DEITY and DAR&PG boosted the e-Governance process.....(**PYQ**, Fullform)
- **DEITY** : **D**epartment of **E**lectronics and **I**nformation **T**echnolog**y**
- **DAR&PG** : **D**epartment of **A**dministrative **R**eforms **and P**ublic **G**rievances
- **IRCTC** : Indian Railway Catering and Tourism Corporation.....**(PYQ)**
- **NIC** stands for _____.**(PYQ, Sample Paper of 2025)** Answer : **N**ational **I**nformatics **C**entre
- **Key Central Initiatives:** NeGP, National e-Governance Division (NeGD)**(PYQ)**, e-Governance Infrastructure, Mission Mode Projects, Citizens/Business/Government Services, Projects/Initiatives, and R&D in e-Governance.
- State governments also launched e-governance initiatives in areas like agriculture, commercial taxes, land records, police, road transport, and treasuries.

B. E-Governance Sites

1. **`india.gov.in` (The National Portal of India):** Provides **single window access** to information and services from various Indian Government entities (central, state, district level). It is a Mission Mode Project under NeGP, maintained by NIC, DeitY, MoCIT, Government of India. **(PYQ(3))**
2. **`goidirectory.nic.in` (Government of India Web Directory):** A one-point source to access all Indian Government websites at all levels and sectors. Includes sites of states/UTs, judiciary, legislature, and sectors like education and agriculture.

IV. Online Shopping and Bill Payments

- **Online Shopping:** An e-commerce application allowing customers to purchase goods over the Internet.....**(PYQ)**
- Customers can view goods, select items, add them to an **online shopping basket** (or cart), make payments (debit/credit card, etc.), and have the goods delivered to the specified address.
- **Examples of Products:** Books, TV, mobile phones, dresses, cosmetics, jewelry, greeting cards.

A. Benefits of Online Shopping.....(PYQ)

- **Convenience:** Shopping can be done anytime, anywhere (no fixed hours).
- **Saves Time/Money:** Eliminates time and cost spent traveling to the store.
- **Access to Unavailable Products:** Allows purchase of products not available locally.
- **Direct Delivery:** Enables sending products directly to a friend's address, saving delivery effort.
- **Bargain Offers**.
- **Customer Ratings/Reviews:** Allows customers to view ratings and reviews of services and items, helping assess quality and popularity.

B. How it Works (Process Overview)

1. **Open the site**.
2. **Browse** (if just looking) or **Login/Signup** (if buying).
    - First-time users fill a registration form (username/password).
    - Some sites allow purchasing without an account or logging in via external accounts (Facebook/Google).
3. **Login** with credentials.
4. **Select products** and **add them to the shopping cart/basket**.
5. **Review the order:** View selected products and the total amount.
6. **Modify or Proceed to Payment:** Users can delete products from the cart or click `<Place order>`.
7. **Read terms and conditions** before payment.
8. **Payment:** Choose a payment option (credit/debit card, net banking, cash on delivery, e-gift voucher, wire transfer).
9. **Complete the transaction**.

- **Examples of Online Shopping Stores:** Flipkart, Ebay, yebhi.com, myntra.com, ibibo.com, snapdeal.com, infibeam.com. **(PYQ)**

C. Bill Payments

- Payments are typically made using debit or credit cards.
- Alternative payment methods: Net banking, e-gift voucher, cash on delivery, and wire transfer.
- Bill payments can be made for government services such as electricity, water bill, property tax, and income tax.
- Precautions for online transactions (related to passwords and fraud) are the same as those required during online reservation.

V. Online Courses, Tutorials, and Tests

- Online learning involves students and teachers at physically different locations connected via the Internet.

A. Benefits of Online Learning.....(PYQ(2), 4 marks)

- **Freedom:** Learners can learn at their convenient location, time, and pace.
- **Non-Discrimination:** No discrimination based on age, caste, gender, background, or qualification.
- **Enriched Content:** Tutorials often include audio and video information, making learning more enjoyable.
- **Variety:** Tutorials and tests are available for almost all topics, sometimes in regional languages.

  Mnemonic: **FENV**

B. Types of Learning Interactions

- **Video/Slide Tutorials:** Lectures available as video or slides for viewing at user convenience.
- **Interactive Tutorials:** Require users to follow instructions, perform actions, and receive feedback.
- **Webinars:** Real-time lectures where users can participate and ask questions. **(PYQ(2),** Give one difference between a webinar and a video tutorial.**)**

C. Online Educational Sites (PYQ (2))

1. **Khan Academy:** Hosts video tutorials on subjects like mathematics, history, physics, and economics.
2. **Udacity (`udacity.org`):** Offers online courses for technical skill development (projects often built by technical leaders like Google). Provides verified certificates for a fee.
3. **Coursera (`coursera.org`) and edx.org:** Provide free, high-quality online courses in collaboration with global universities (humanities, science, engineering, school level courses). Offers verified certificates for some courses for a fee.
4. **w3schools.com:** Offers free tutorials in web development technologies (HTML, CSS, Javascript, PHP, etc.).
5. **GCFLearnFree.org:** Provides free online learning opportunities to improve technology, literacy, and math skills.
6. **NCERT Portal (`ncert.nic.in`):** Provides online learning resources (e-books, journals, question papers).

Note : CBSE Latest Marking Scheme (24-25) Also Recognizes ‘Unacademy’ and ‘physicswallah’ as **valid** answers.

D. Online Testing Sites

- Websites host tests in different subjects, allowing learners to take tests and get evaluated.
- **Example (Free):** `british-study.com/quizzes/grammar-test.php` for English grammar.
- **Example (Paid):** `pearsonvueindia.com/lsatindia/...` for standardized tests like LSAT—India (Law School Admission Test).

VI. Project Management - Web Based Application Development

- **Web Application:** A program stored on a **remote** server **(PYQ(2), Definition and short q)**, delivered over the Internet through a web browser interface (e.g., quiz, game, bill calculator).
- **Project:** A task undertaken to create a unique product, service, or result.
- **Key Characteristics of a Project**: **(PYQ)**
    - Defined **beginning and end** (boundaries).
    - Requires **finite resources**.
    - Has a **specific time frame** (definite beginning and end dates).
    - Objectives are **specific and identifiable**.
- **Project Management:** The process of achieving project objectives through defined activities within a time frame, using knowledge, skills, tools, and techniques.

A. Phases in a Web Application Project

The structured development process is broadly categorized into four main phases:

1. **Requirements Definition Phase (PYQ)**
2. **Design Phase** .....(**PYQ**, Fill in the blanck)
3. **Implementation Phase** (**PYQ (3)**)
4. **Testing Phase (PYQ)**

- **Planning and Monitoring** occurs continuously throughout all phases, tracking aspects like duration, costs, schedule, resources, and risk.

B. Detailed Phases and Steps

Phase	Focus/Goal	Steps and Content
1. **Requirements Definition**	Identifying the problem statement and defining the "what" the application must do. **(PYQ)**	**Feasibility Check:** Assess technical feasibility (is it possible?) and economic feasibility (is it profitable?). **Scope Definition:** Compile a detailed list of application features, establish goals, and recognize limitations placed on the project. **(PYQ)**
2. **Design**	Focusing on the "how" of implementation.	**A Map of the Web Application Site:** Define the structure (site map), pages, and relationships between them. **Database Design:** Identify necessary tables, attributes, data types, column sizes, and relationships between tables. **Page Structure Design:** Identify main page sections (header, main content, sidebar).
3. **Implementation**	Creating the functional components.	**Backend Database:** Write SQL code to define the database, tables, attributes, and relationships.....(**PYQ**, RDBMS is frontend/backend) **Frontend:** Develop the user interface (UI) based on page structure/layout.....(**PYQ**, Define Frontend wrt web based applcn) Design input controls (Radio buttons, checkboxes, list/combo boxes) to minimize user typing effort. **Data Connectivity:** Establish connectivity between the Front-End interface and the Back-End Database.....**(PYQ)**
4. **Testing**	Validation and Quality Assurance.	Test the complete application (Front-End and Back-End) using multiple sample data sets. Find and fix application bugs. **(PYQ)** Validate modified code after fixing errors. (Application is ready for release after final validation).

C. Case Study Examples (Design Highlights)

Case Study	Problem/Functionality	Key Entities (Tables) Identified
Online Game (Hangman)	Player guesses a word from a category with limited chances (7 misses).	`Category` (Category_Id, Category_Name). `Words_in_Category` (Category_Id, Word).
Online Quiz	Allows users to register, login, select a quiz category, attempt questions, and automatically generate results. Administrator manages questions.	`Admin_Login` (Username, Password). `Quiz_Info` (Question_No, Subject, Question, Options 1-4, Answer). `User_Info` (Username, Password, Name, Email_id, Gender, Dob). `Result` (Username, Marks, Category).
Online Bill Calculator	Allows users to register, login, enter bill details, and generate a bill. Administrator manages user accounts.	`User_Info` (Username, Password, Name, Address, City, State, Country, Pin_Code, Mobile_Number). `Bill_Info` (Username, Name, Address, City, State, Country, Pin_Code, Month_Dispatch, Duration of Bill, Mobile_Number).

UNIT III: Java Programming

I. Introduction and Program Execution

1. **Computer Program Definition:** A computer program is a sequence of instructions provided to the computer to accomplish a specific task.
2. **Programming:** This refers to the preparation and writing of programs for computers.
3. **High-Level Languages:** Programmers write programs in high-level programming languages such as Java, C++, Python, Ruby, or Scala.
4. **Compiler Function:** A compiler is necessary to translate **high-level program code** into **machine language code**, which is the only language a computer understands.....(**PYQ**, A computer can only understand _____) (**PYQ**, What is a compiler?)
5. **Java Compiler Output:** A Java compiler translates Java code not directly into machine language, but into **Java Bytecode** (a highly optimized set of instructions.....**(PYQ(2), SQP 2025- 2026)**). The bytecode is also referred to as a **Java class file**.
6. **Java Virtual Machine (JVM):** When the bytecode is run on a computer, a Java interpreter called the **JVM** translates the bytecode into machine code and then executes it. .....(**PYQ(4)**)
7. **Platform Independence:** The JVM approach allows compiled Java **bytecode** to be run on **any platform** (like Windows, Linux, or Mac) that has a JVM running on it. This makes Java programs platform independent and highly portable. **(PYQ(2))**
8. **Development Environment:** To write Java programs, one needs a Text Editor (for writing code) and a Java compiler. **Integrated Development Environments (IDEs)**, such as the open-source and free **Java NetBeans IDE**, combine these tools, simplifying writing, compiling, and executing programs. **(PYQ)**

II. Basic Program Structure and Components

1. **Project Organization:** Software development in NetBeans is organized in the form of **Projects**.

2. **File Naming Convention:** Java requires that the name of the class should be the **same as the source file name** (e.g., `HelloWorld` class in `HelloWorld.java`).
3. **Packages:** A package in Java is a group of related classes. The first line in a Java program is often a package statement declaring which package the program belongs to. All classes within a package can share their data and code.
4. **Classes:** Class declarations begin with the keyword `public class` followed by the class name, and their contents are enclosed within curly braces `{}`.
5. **Methods:** A method is a group of statements written to perform a specific task. The method body is enclosed within curly braces.... (**PYQ(3)**, As it is line copy pased with [method] blanck)
6. **`main` Method:** The `main` method is a special method that **every Java application must have**. When a program runs, the statements in the `main` method are the first ones executed.
7. **Statements:** All Java statements must end with a **semicolon** (`;`).....**(PYQ)**
8. **Comments:** Comments are used by programmers to document their programs and are ignored by the compiler.
    - **Single-line comments** start with two consecutive forward slashes (`//`).
    - **Multiple-line comments** are written between the symbols `/*` and `*/`. (**PYQ**, Multiple line comment) (**PYQ**, Two ways to write comment)
9. **Output Methods:** Java provides pre-built output methods to display information:
    - `System.out.println()`: Prints the output followed by a **newline** character.
    - `System.out.print()`: Prints the output without a newline character, causing subsequent output to be printed on the **same line**.

III. Data Types and Variables

1. **Variables Definition:** A variable is a **placeholder for data** that can change its value during program execution. **(PYQ(2))** Technically, a variable is the name for a storage location in the computer's internal memory, and its value is the contents at that location.
2. **Variable Declaration:** All data variables in Java must be declared and initialized before they are used. (**PYQ**, True or False) Declaration requires specifying the data type to tell the compiler how much memory to reserve.
3. **Local Variables:** These variables are available only inside the methods where they are declared.
4. **Case Sensitivity:** Java is a **case-sensitive** language; thus, variable names written in different cases are treated as distinct variables (e.g., "percentage" differs from "PERCENTAGE").
5. **Variable Naming Rules: (PYQ)**
    - They can begin with an alphabetic character, an underscore (_), or a dollar sign ($). Conventionally, they start with a letter.
    - They can only consist of alphabets, digits, and underscores.
    - They must be a single word (spaces are not allowed, but underscores are).
    - Reserved words (like `int`) cannot be used as variable names.
    - It is good practice to make variable names meaningful.
6. **Multiple Declarations:** Multiple variables of the same type can be defined in one statement, separated by commas (e.g., `int num1, num2, num3;`).
7. **Primitive Data Types:** Java supports **eight** primitive data types: **(PYQ)**
    - **Integer Types:** `byte` (8-bit), `short` (16-bit), **`int`** (32-bit), `long` (64-bit).
    - **Floating Point Types:** `float` (32-bit), **`double`** (64-bit).
    - **Character Type:** **`char`** (16-bit, stores a single character, assigned using single quotes, e.g., `'M'`).
    - **Boolean Type:** **`boolean`** (1-bit, stores `True` or `False`).
8. **String Variables:** To store textual data consisting of more than one character, the **`String` class** (not a primitive type) is used. Text values are assigned by enclosing the text between **double quotes** (e.g., `"Mayank"`).

IV. Operators

Operators are special symbols that perform specific operations in a programming language.

1. **Arithmetic Operators**:
    - `+`: Addition
    - `-`: Subtraction
    - `*`: Multiplication
    - `/`: Division (returns quotient)
    - `%`: Modulus (returns remainder)
    - `++`: Increment (increments operand by 1)
    - `--`: Decrement (decrements operand by 1)
2. **Relational Operators** (used to form conditions/expressions): **(PYQ)**
    - `==`: Equal to
    - `!=`: Not equal to
    - `>`: Greater than
    - `<`: Less than
    - `>=`: Greater than or equal to
    - `<=`: Less than or equal to
3. **Assignment Operators**:
    - `=`: Simple Assignment

- `+=`: Add and Assignment (e.g., `a += b` is same as `a = a + b`)
- `-=`: Subtract and Assignment
- `*=`: Multiply and Assignment
- `/=`: Divide and Assignment
- `%=`: Modulus and Assignment

4. **Logical Operators**: **(PYQ(2))**
    - `&&`: Logical AND
    - `||`: Logical OR
    - `!`: Logical NOT
5. **Concatenation Operator:** The `+` operator performs addition when used with numbers as operands, but performs **concatenation** when used with Strings. **(PYQ)**

V. Control Flow (Selection and Repetition)

Java executes instructions sequentially, but control flow statements allow conditional execution (selection) or repeated execution (repetition/looping).

Selection Structures (Conditional Execution)

1. **`if` Statement:** Executes a block of code (sequence of statements in curly braces) only if a conditional expression (condition) evaluates to `true`.
2. **`if else` Statement:** Executes the block following `if` if the condition is true, or executes the block following `else` if the condition is false. `if-else` statements can be nested.
3. **`switch` Statement:** Used to execute a block of code matching one value out of many possible values.....**(PYQ(2))**
    - The **expression** in the `switch` statement must evaluate to `byte`, `short`, `int`, or `char`.
    - The expression is compared with the constant values defined in multiple `case` groups.
    - If no match is found, statements in the **`default`** part are executed.
    - The **`break`** statement is used after each case group to terminate the switch execution.

IMPORTANT PYQ ON SWITCH CASE

QUESTION 1:

What will be the final output shown in the text field when a = 1?

```
switch (a)
{
    case 0: jTextField1.setText("Blue");
    case 1: jTextField1.setText("Red");
    case 2: jTextField1.setText("Orange");
            break;
    case 3: jTextField1.setText("Black");
    default: jTextField1.setText("transparent");
             break;
}
```

Answer:

OUTPUT : "Orange" (Due to fall through)

Modified QUESTION 1:

If the same switch statement used System.out.println() instead of setText(),

what would be printed on the console when a = 1?

```
switch (a)
{
    case 0: System.out.println("Blue");
    case 1: System.out.println("Red");
    case 2: System.out.println("Orange");
            break;
    case 3: System.out.println("Black");
    default: System.out.println("transparent");
             break;
}
```

Answer:

OUTPUT: Red

Orange

Repetition Structures (Looping)

1. **Looping:** The ability of a computer to perform the same set of actions repeatedly until some condition is met.
    - **Body of the loop:** The sequence of statements repeated.
    - **Iteration:** A single pass through the loop.
2. **`while` Statement:**
    - An **entry controlled loop** (it evaluates the test condition *before* executing the loop body).
    - The loop executes **zero or more times**.

IMPORTANT PYQ

Q) Which of the following is/are not a keyword in JAVA ?
break, While, for, switch, Display

```
Answer : While, Display. (While isnt a real keyword in java because its w is capitalised. In java while isnt capitalised
```

3. **`do while` Statement:**
    - An **exit control loop** (it evaluates the test condition *after* executing the loop body).
    - The loop body executes **at least once**. **(PYQ)**
    - A semicolon (`;`) must be placed at the end of the test condition (e.g., `} while (expression);`).

PYQ : Differentiate between Entry controlled loop and Exit controlled loop.
PYQ : 16. Differentiate between while loop and do while loop in Java.

4. **`for` Statement:** The most widely used loop construct. It has three parts separated by semicolons:
    - **Initialization:** Sets the starting value of the loop counter.
    - **Test Condition:** Determines whether the loop should execute again (loop is exited when the condition fails).
    - **Change Counter (Step):** Updates the counter in each loop iteration.
    - The loop counter can be incremented (counting up) or decremented (counting down) by any value.

PYQ : Write a Java program to calculate and display the factorial of a number 5 using for loop.

VI. Arrays

1. **Array Definition:** Arrays are variables that can hold a **list of values of the same type**. **(PYQ(2))**
2. **Creation:** An array is declared using square brackets (`[]`) after the data type (e.g., `double[] marks;`). The size of the array is specified using the `new` keyword (e.g., `marks = new double;`).
3. **Indexing:** Array elements are indexed by their position, starting from **0**. An array of size *n* has indices from 0 to *n-1*.
4. **`length` Property:** An array variable has a useful property, `length`, which gives the size of the array (e.g., `marks.length`).
5. **Error Condition:** Referencing an index outside the valid range (0 to length-1) causes an **Array index out of bounds error**.

VII. User Defined Methods

1. **Method Structure:** A method has a name, a return type, an optional list of parameters, and a body.
2. **Parameters:** The parameter list (within round brackets) allows data to be passed into the method. The parameters act as local variables inside the method body.
3. **Return Type:** The `return type` is the type of the value that the method returns back to the calling method.
4. **`void` Return Type:** If a method does not return any value, the special return type **`void`** is used.
5. **Method Invocation:** A method is called or invoked from another method. Arguments supplied during the call are copied into the parameters of the called method.
6. **`static` Modifier:** The `static` modifier allows a method to be called without requiring an invoking object.

VIII. Object Oriented Programming (OOP) and Class Design

1. **OOP Language:** Java is an **Object Oriented Programming (OOP)** language. In OOP, a program is a collection of objects that interact with each other to solve a problem.....(**PYQ, 4 marks**, Why is Java considered as an Object Oriented Programming language ? Explain data members and member methods with the help of an example. Ans : cuz it adhere to core OOP principle like encapsulation, polymorphism, data abstraction etc.)
2. **Class:** A class is a physical or logical entity that serves as a **template** from which objects are created. A class has certain characteristics or attributes. (**PYQ,** According to Object-Oriented Programming Language, a/an _____ is a physical or logical entity that has certain attributes)
3. **Object:** An object is an **instance of a class**. To describe a specific entity (e.g., a book), an object of the class is created and populated with data.
4. **Members:**
    - **Data Members (Attributes):** Variables within a class that define the characteristics of an object. They are like global variables that can be accessed by all method members of the class. **(PYQ)**
    - **Method Members:** Functions within a class that perform actions on the data members (e.g., display or calculate). **(PYQ)**
5. **Accessing Members:** Data and method members of an object are accessed using the **dot (`.`) operator**. (**DPS** Post Mid Term, Name the operator used to access the members of a class.)
6. **Base Class:** All Java classes inherit from the base class **`Object`**.
7. **Constructors:**
    - A special method used to initialize the data members of a class.

- It has the **same name as the class**.
- It has **no return type**.
- It is invoked automatically whenever a new object of the class is created using the `new` keyword.
- A class can define multiple constructors, including a parameter-less constructor (one with no arguments).

Access Control and Modifiers (**PYQ** : Explain the access modifiers and their types in Java.)

1. **`private` Access Modifier:** When a member (data or method) is declared `private`, it is visible **only within the class** and cannot be accessed from outside. This is considered good programming practice for data members to maintain control over their values and avoid security issues.....(**PYQ (3)**, Fill in the blanck)default
2. **`public` Access Modifier:** A `public` member has visibility both **within and outside the class**. By default, all members of a class are public.
3. **Getter and Setter Methods:** These methods provide controlled access to private data members from outside the class.
    - **Getter Method:** Returns the value of a private data member.
    - **Setter Method:** Allows modification of a private data member, often including control logic (e.g., restricting a price from falling below a minimum value).

IX. Java Libraries and Utilities

The power of Java stems from the hundreds of prebuilt classes available in its libraries.

1. **Importing Libraries:** To use a prebuilt class, the `import` keyword must be used to import the class from its package into the current program space. Import statements must appear before any class definitions.....**(PYQ)**
2. **Data Input (Using `java.util.Scanner`):(PYQ, In which Java package is the scanner class available ?)**
    - The `Scanner` class is used to take user input.
    - A `Scanner` object is created using `System.in` as the source (e.g., `Scanner user_input = new Scanner(System.in);`).
    - The `next()` method of the `Scanner` class returns the input token as a `String`.
    - **Wrapper Classes** (e.g., `Integer`, `Double`) contain static methods (like `parseInt()` or `parseDouble()`) to convert input `String` into the required numeric data type. (**DPS Post Mid Term Question**, Define the term Wrapper class. Answer : A *wrapper class* is a Java class that **wraps a primitive data type inside an object**, allowing the primitive to be handled **by reference instead of by value**.)
    - A wrapper class in Java is a class that encapsulates a primitive datatype inside an object, allowing the value to be treated like an object and accessed through class methods. So it just means that variables point to the address of the object which has the value so both the caller and the method refer to the same object. but Primitive variables store values directly and when passed to a method only a copy of the value is passed, so the original cannot be changed.

(**PYQ**, Wrapper
The statements given below use Wrapper class to store an integer value, 100 and then display it. Fill in the blanks to complete the statements.
Integer num = _____ (100);
System.out.println (_____);)

Answer:
Integer num = new Integer(100);
System.out.println(num);

Good Preboards Question

```
String s1 = "Hello";
String s2 = new String("Hello");
System.out.println(s1 == s2);
```

Answer:
false

3. **Array Manipulation (Using `java.util.Arrays`):**
    - The `Arrays` class provides useful methods for array handling.
    - `sort()`: Sorts an array of numbers in ascending order or an array of Strings in alphabetic order. **(PYQ)**
    - `binarySearch()`: Searches for a specific key element; it returns the index of the element if found, or -1 otherwise. **(PYQ)** **The array must be sorted before calling `binarySearch()`.**
4. **String Manipulation (Using `java.lang.String`):**
    - The `String` class provides numerous methods for manipulating text.
    - Examples include `toUpperCase()`, `toLowerCase()`, `length()`**`(PYQ)`**, `charAt(index)`, `indexOf(str)`, `concat(str).....(PYQ (2),Concat)`, `replace (oldChar, newChar)..(PYQ)`, and comparison methods like `equals()` and `equalsIgnoreCase()`.

X. Advanced Java Features

Exception Handling

1. **Exception Definition:** An exception is an unexpected error situation during program execution (e.g., dividing an integer by zero, or accessing an array index out of bounds) that causes the program to terminate unexpectedly. **(PYQ)**
2. **Purpose:** Java's exception handling allows a program to deal with these errors, continue executing, or terminate gracefully...(**PYQ**, 2 marks , What is the purpose?)
3. **Exception Handling Keywords**:
    - **`try`:** Surrounds the part of the code where an exception might occur.
    - **`catch`:** Follows a `try` block and contains the exception handler (specific code executed when an exception occurs).... **(PYQ(2))** The `catch` argument must match the type of the exception object thrown. The `Exception` class is a general "catch all" type.
    - **`finally`:** An optional block that is **always executed** when the `try` block exits, regardless of whether an exception occurred or not. It is typically used for clean-up code (e.g., freeing up resources).
4. **Retrieving Error Message:** The `getMessage()` method of the `Exception` class can be used to print the error that caused the exception.

The **types of errors/abnormal conditions** that can be handled using exception handling in Java are:

- **Array index out of range**
- **Division by zero**
- **Insufficient computer memory**

```
class AdvancedJavaExceptionDemo {
    public static void main(String[] args) {
        int[] numbers = {10, 20, 30};
        int result = 0;

        try {
            // 1. Exception scenarios:

            // Array index out of range
            System.out.println("Accessing 5th element: " + numbers[5]);

            // Division by zero
            result = numbers[0] / 0; // This line won't execute because previous line throws an exception

            // Simulate insufficient memory (just for demo, won't actually crash)
            int[] bigArray = new int[Integer.MAX_VALUE];
        }
        catch (ArrayIndexOutOfBoundsException e) {
            System.out.println("Array Error: " + e.getMessage()); // Retrieve error message
        }
        catch (ArithmeticException e) {
            System.out.println("Math Error: " + e.getMessage());
        }
        catch (OutOfMemoryError e) { // Example of handling memory issue
            System.out.println("Memory Error: " + e.getMessage());
        }
        catch (Exception e) { // General catch-all exception
            System.out.println("Some Other Error: " + e.getMessage());
        }
        finally {
            System.out.println("Finally block executed: Clean-up can be done here.");
        }

        System.out.println("Program continues after exception handling.");
    }
}
```

Assertions (**PYQ :**With reference to Java programming language, define assertion. State any one way to write an assertion.)

1. **Purpose:** Assertions are mechanisms for effectively identifying, detecting, and correcting **logical errors** in a program during development/debugging.
2. **Function:** An `assert` statement specifies a condition that should be true at a particular point during execution. If the condition is false, an `AssertionError` is thrown.
3. **Enabling Assertions:** Assertions reduce runtime performance, so they are **disabled by default**. They must be explicitly enabled **at runtime** using the **`-ea`** VM option.

   java -ea [class name]

```
class AssertionExample {
    public static void main(String[] args) {
        int age = 15;

        // 1 & 2. Purpose & Function:
        // This assertion checks a logical assumption: age should be >= 18
        // If false, AssertionError is thrown
        assert age >= 18 : "Logical Error: Age must be 18 or above";
```

```
        System.out.println("Program continues if assertion passes.");
    }
}
```

Threads

1. **Definition:** A multithreaded program is one that can perform multiple tasks **concurrently** (at the same time), optimizing resource utilization. The multiple parts are called threads.
2. **Thread Creation Methods: (PYQ(2), 2 mark)**
    - **Extending the `Thread` class** (and overriding the `run()` method).
    - **Implementing the `Runnable` interface** (and overriding the `run()` method). This method offers more flexibility because the class can still extend other classes.
3. **Execution:** The `run()` method is the entry point for every new thread. Execution is started by calling the `start()` method of the `Thread` class.
4. **Control:** Thread priority levels can be set using `setPriority()` (ranging from 1 to 10). The `sleep()` method causes a thread to suspend execution for a specified number of milliseconds.

```
// 1. Thread creation by extending Thread class
class MyThread extends Thread {
    public void run() {  // 3. run() is entry point
        for (int i = 1; i <= 3; i++) {
            System.out.println("Thread from Thread class: " + i);
            try {
                Thread.sleep(500); // 4. pause execution for 500ms
            } catch (InterruptedException e) {
                System.out.println(e.getMessage());
            }
        }
    }
}

// 2. Thread creation by implementing Runnable
class MyRunnable implements Runnable {
    public void run() {  // 3. run() is entry point
        for (int i = 1; i <= 3; i++) {
            System.out.println("Thread from Runnable: " + i);
            try {
                Thread.sleep(500); // 4. pause execution
            } catch (InterruptedException e) {
                System.out.println(e.getMessage());
            }
        }
    }
}

public class ThreadDemo {
    public static void main(String[] args) {
        // Create thread by extending Thread
        MyThread t1 = new MyThread();
        t1.setPriority(Thread.MAX_PRIORITY); // 4. set thread priority
        t1.start(); // 3. start execution

        // Create thread by implementing Runnable
        Thread t2 = new Thread(new MyRunnable());
        t2.setPriority(Thread.MIN_PRIORITY); // 4. set priority
        t2.start(); // 3. start execution

        System.out.println("Main thread is running concurrently...");
    }
}
```

Wrapper Classes

1. **Purpose:** Wrapper classes are provided by Java to wrap primitive datatypes (like `int`, `float`) into objects. This allows primitive types to be treated as objects, which is necessary if they need to be passed by reference.
2. **Examples:** `Integer` for `int`, `Double` for `double`, `Boolean` for `boolean`, etc.. These classes are in the `java.lang` package and imported by default.
3. **Methods:** They provide utility functions, such as:
    - `intValue()`: A getter function to access the primitive value held in the object.
    - `Integer.parseInt()`: A static method to convert a String into its integer equivalent.
    - `Integer.toString()`: Converts an integer value to a String.

```
public class WrapperDemo {
    public static void main(String[] args) {

        // 1. Purpose: Wrapping primitives into objects
        int num = 42;
        Integer wrappedNum = Integer.valueOf(num); // wrap int into Integer object
        System.out.println("Wrapped Integer: " + wrappedNum);

        // 3. Methods

        // a) intValue(): get primitive back from object
        int primitiveNum = wrappedNum.intValue();
        System.out.println("Primitive value: " + primitiveNum);
```

```
        // b) parseInt(): convert String to int
        String str = "123";
        int parsedNum = Integer.parseInt(str);
        System.out.println("Parsed integer from String: " + parsedNum);

        // c) toString(): convert int to String
        String strFromInt = Integer.toString(primitiveNum);
        System.out.println("Integer converted to String: " + strFromInt);

        // 2. Examples of other wrapper classes
        Double doubleObj = Double.valueOf(3.14);
        Boolean boolObj = Boolean.valueOf(true);
        System.out.println("Wrapped Double: " + doubleObj);
        System.out.println("Wrapped Boolean: " + boolObj);
    }
}
```

XI. Database Connectivity

Java allows connectivity to databases, such as MySQL, through the use of specific libraries and methods.

1. **NetBeans Configuration:** The MySQL Server must first be registered and connected within the NetBeans IDE Services tab, requiring the Administrator username and password.
2. **MySQL Connector JAR:** Before writing connection code, the appropriate MySQL connector JAR file (e.g., `mysql-connector-java-5.1.23-bin.jar`) must be added to the project's Libraries node.
3. **Required Package:** Classes necessary for database operations reside in the **`java.sql`** package.
4. **Establishing Connection:**
    - The `DriverManager.getConnection()` method is invoked to establish a database connection.
    - It requires the database **URL** (including host, port 3306, and database name), username, and password.
    - It returns a **`Connection`** object.
5. **Executing Queries:**
    - The `Connection.createStatement()` method returns a **`Statement`** object for sending SQL statements.
    - The `Statement.executeQuery(query)` method executes an SQL query and returns a single **`ResultSet`** object.
6. **Processing Results:**
    - The `ResultSet.next()` method is used to iterate through all rows of data returned by the query. It returns `false` when no more rows are left.
    - Data is retrieved from columns using methods like `getString(index)`.
7. **Error Handling:** All database connection and fetching statements must be wrapped in a `try-catch` block to handle **`SQLException`**.

Q Create a java class "student" with private data members aname and marks along with getter and setter methods to access and modify them

```
class Student {
    // private data members (encapsulation)
    private String aname;
    private int marks;

    // getter for aname
    public String getAname() {
        return aname;
    }

    // setter for aname
    public void setAname(String aname) {
        this.aname = aname;
    }

    // getter for marks
    public int getMarks() {
        return marks;
    }

    // setter for marks
    public void setMarks(int marks) {
        this.marks = marks;
    }
}
```